(71)

$\Omega(n^7)$

$1n^7 - 5n^6 - 10n^3 + n^2 - 1 \geq \frac{1}{2} n^7$

$c_1 = \frac{1}{2}$

$$n_0 = \frac{\sum_{i=0}^{k-1} |a_i|}{a_k - c} = \frac{5+10+1+1}{1-\frac{1}{2}}$$

$$n_0 = \frac{17}{\frac{1}{2}} = 34$$

$O(n^7)$ $n^7 - 5n^6 - 10n^3 + n^2 - 1 \leq 18 n^7$

$O(n^7)$ $c_2 = 18$, $n_0 = 1$

$\therefore \Theta(n^7)$, $c_1 = \frac{1}{2}$, $c_2 = 18$, $n_0 = 34$

(11)

third method

Master Theorem

if $\boxed{T(n) = a \cdot T(n/b) + f(n)}$

فإنه حلها كما يلي

لاحظ معطيات النظرية

(1) $a \cdot T\left(\frac{n}{b}\right) + f(n)$

ضرب

لابد من أنه تكون قسمة

دالة في (n)

$a > 1 \; , \; b > 1$

تكون حل T(n) كما يلي

(i) أكبر

(ii) أصغر

(iii) نقارن ونتيجة

$\log_b a$

$n^{\log_b a}$

$\boxed{n^{\log_b a} \quad \text{with} \quad f(n)}$

3 cases.

د/مصطفى 0508783424

Scanned by CamScanner

case1 : $n^{\log_b a} > f(n)$

في الدرجة

$$T(n) = \Theta\left(n^{\log_b a} \cdot \cancel{\log}\right)$$

Case 2 : $f(n) = n^{\log_b a}$

لها نفس الدرجة

$$T(n) = \Theta\left(n^{\log_b a} \cdot \log n\right)$$

Case3 $n^{\log_b a} < f(n)$ and

$af(n/b) < c f(n)$ for large n

و $\boxed{c < 1}$

$$T(n) = \Theta\left(f(n)\right)$$

Ex ① solve $T(n) = 9T(\frac{n}{3}) + n$

sol

$a = 9 \quad , \quad b = 3 \quad , \quad f(n) = n$

$3^{\cdots} = 9$

(i) $\log_b a = \log_3 9 = 2$

(ii) $n^{\log_b a} = n^2$

$\because n^2 > n$ case 1

$T(n) = \Theta(n^{\log_b a}) = \Theta(n^2)$

ملاحظة : log نستخدم لوغاريتم

$\log_{10} 100 = 2 \Rightarrow 10^2 = 100$

$\log_{10} 1000 = 3 \Rightarrow 10^3 = 1000$

$\log_2 32 = 5 \Rightarrow 2^5 = 32$

Ex 2) $T(n) = 2T\left(\frac{n}{2}\right) + n^3$

sol

$a = 2$, $b = 2$, $f(n) = n^3$

(i) $\log_b a = \log_2 2 = 1$

(ii) $n^{\log_b a} = n^1$

$n < n^3$ case 3

$f(n) = n^3$

$f(A) = A^3$

$f(x) = x^3$

$f\left(\frac{n}{2}\right) = \left(\frac{n}{2}\right)^3$

$a f(n/b) < c f(n)$

$2 \cdot f\left(\frac{n}{2}\right) < c \cdot n^3$

$2 \cdot \left(\frac{n}{2}\right)^3 < c \cdot n^3$

$2 \cdot \frac{\cancel{n^3}}{8} < c \cdot \cancel{n^3}$

$2 \cdot \frac{1}{8} < c \Rightarrow \cancel{\frac{1}{4}} < c$

$\Rightarrow c > \frac{1}{4} \Rightarrow c < 1$

case 3

$$T(n) = \Theta(f(n))$$

$$T(n) = \Theta(n^3)$$

Ex 3 $\quad T(n) = 4T\left(\frac{n}{2}\right) + 2n^2$

sol

$a = 4 \quad , \quad b = 2 \quad , \quad f(n) = 2n^2$

(1) $\log_b a = \log_2 4 = 2$

$n^{\log_b a} = n^2$

$n^{\log_b a}$ And $f(n)$ لهما نفس الدرجة

Case 2

$$T(n) = \Theta(n^{\log_b a} . \log n)$$

$$\boxed{T(n) = \Theta(n^2 . \log n)}$$

End chap

Using The Master Method

- $T(n) = 9T(n/3) + n$
  - $a=9$, $b=3$, $f(n) = n$
  - $n^{\log_b a} = n^{\log_3 9} = \Theta(n^2)$
  - Since $f(n) = O(n^{\log_3 9 - \varepsilon})$, where $\varepsilon=1$, case 1 applies:

$$T(n) = \Theta\left(n^{\log_b a}\right) \text{when } f(n) = O\left(n^{\log_b a - \varepsilon}\right)$$

  - Thus the solution is $T(n) = \Theta(n^2)$

$n^2 = n^3 - \varepsilon$

Application of Master Theorem

- $T(n) = T(2n/3)+1$ $= T\left(\frac{n}{(\frac{3}{2})}\right)+1$
  - a=1,b=3/2, $f(n)$ =1
  - $n^{\log_b a} = n^{\log_{3/2} 1} = \Theta(n^0) = \Theta(1)$
  - By case 2, $T(n)= \Theta(\lg n)$.

$\cancel{2^{\log_b^a}} * \log n = \log n$

$\log_2 3$

ملاحظة

$2 < 3 < 4$

$\log 2 < \log 3 < \log 4$

$1 < \log 3 < 2$

$n^{\log 8} < n^2$

$n^{\log 3} > n$

Scanned by CamScanner

(B7)

Application of Master Theorem

- $T(n) = 3T(n/4)+n\lg n$;
  - $a=3, b=4, f(n) = n\lg n$
  - $n^{\log_b a} = n^{\log_4 3} = \Theta(n^{0.793})$
  - $f(n)= \Omega(n^{\log_4 3+\varepsilon})$ for $\varepsilon\approx 0.2$
  - Moreover, for large n, the "regularity" holds for c=3/4.
    - $af(n/b) = 3(n/4)\lg(n/4) \le (3/4)n\lg n = cf(n)$
  - By case 3, $T(n) = \Theta(f(n)) = \Theta(n\lg n)$.

$1 < 3 < 4$

$\log_4 1 < \log_4 3 < \log_4 4$

$0 < \quad < 1$

$af(n/b) \le cf(n) \Rightarrow 3(n/4)\log(n/4) \le c\, n\log n$

zero $n^{\log_4 3} < n^1 < n\log n$

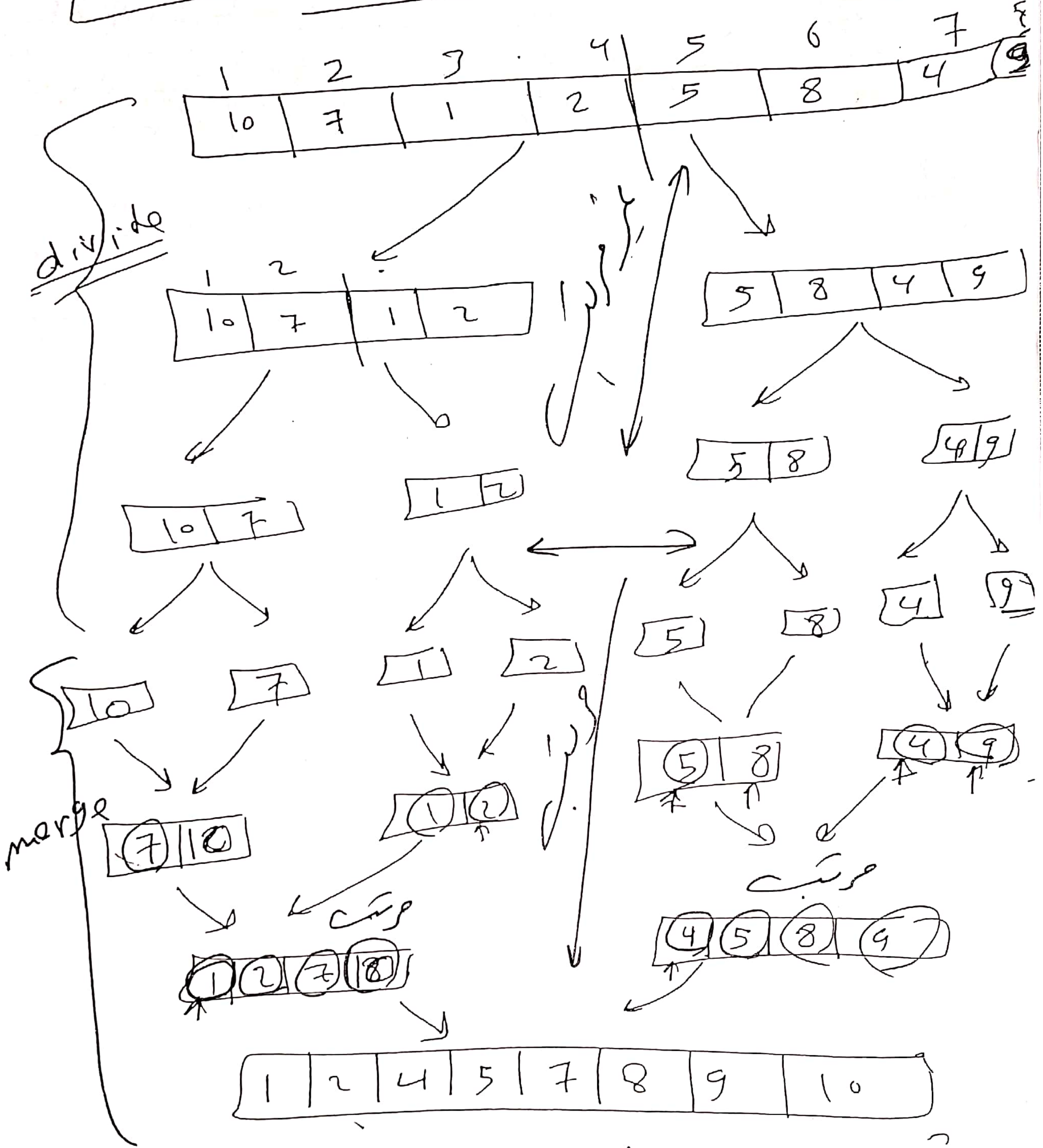
(3)

ch4: Devide and conquer

n log n

Merge sort

Al-nafisah



الكود من ال slides

7/٢٥

40

A

1	2	3	4	5	6	7	8	9	10	11	12	13

merge

بعد عدة خطوات

b: مرتبة

1	2	3	4	5	6
3	5	10	12	13	15

$i = 1$

c: مرتبة

1	2	3	4	5	6	7
1	3	6	8	9	10	11

$j = 1$ $j = 2$

1	2	3	4	5	6	7	8	9	10	11	12	13
1	3	3	5	6								

$k = 1$

procedure merge (b[ ] , c[ ]){

$i \leftarrow 1$ // index b
$j \leftarrow 1$ // " c
$k \leftarrow 1$ // " d

المرحلة الأولى
while ($i \le |b|$ and $j \le |c|$){ true false
d[k++] ← b[i] < c[j] ? b[i++] : c[j++]
الشرط
}

معناها b مازال فيها عناصر ← if ($i \le |b|$){
while ($i \le |b|$) d[k++] ← b[i++]
}

معناها c مازال فيها عناصر
else
while ($j \le |c|$) d[k++] ← c[j++]

}

~~T(n) = b~~ $O(|b| + |c| - 1)$

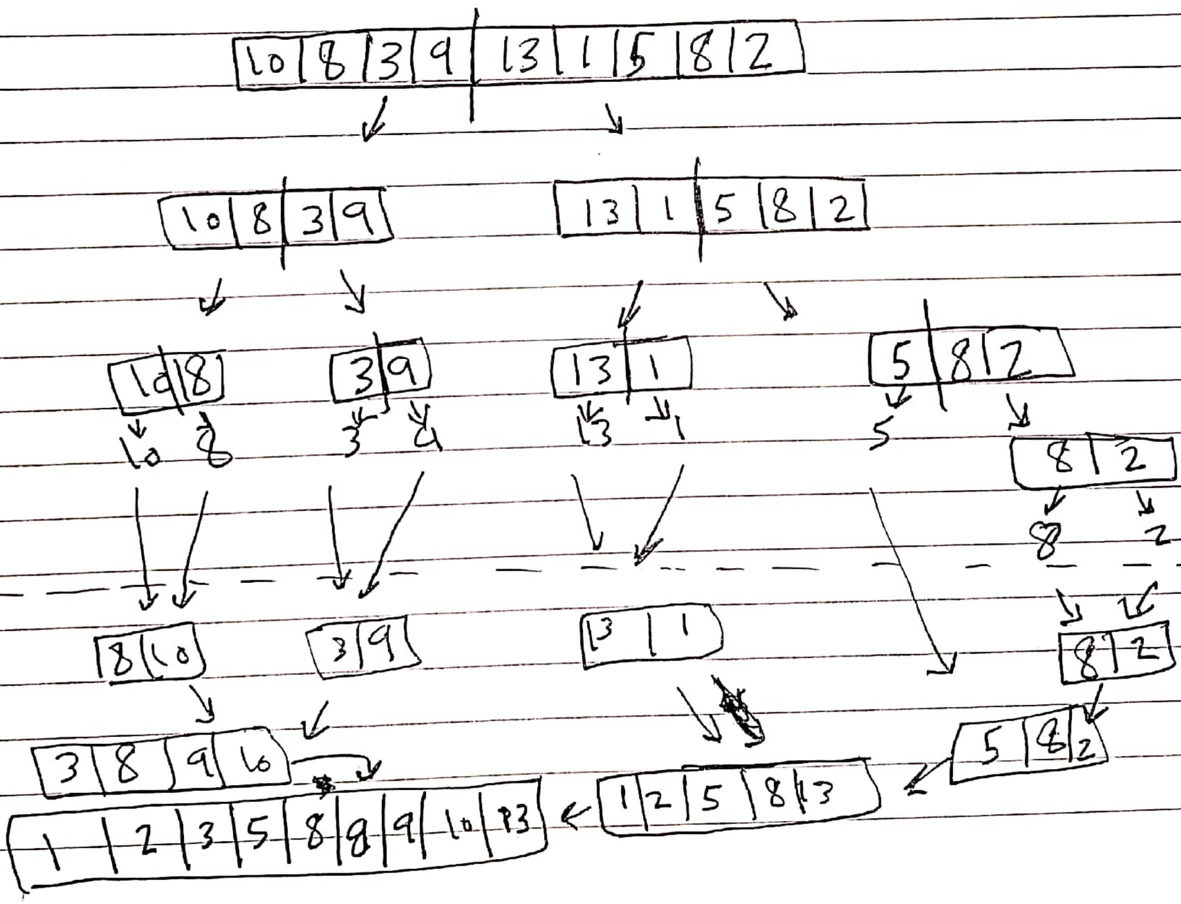
7/٢٥ ص ٤١

mergesort

```
procedure msort (a[], n){            الأول
    if (n>1){
        b ← a[1 ... ⌊n/2⌋] // left half
        c ← a[⌊n/2⌋+1 ..... n] // right half
        a ← merge( msort(b, ⌊n/2⌋),          → T(n/2)
                   msort(c, ⌈n/2⌉) )         → T(n/2)
    }
}
```

مصفوفة ، مرتبة

EX:



Time complexity for merge Function

b

x	y	z	w

c

m	n	o	p

d

x	m	y	n	z	o	w	p

دمج مصفوفة مرتبة

$\therefore$ Time $= \overbrace{|b|+|c|}^{n-1} - 1 = O(|b|+|c|) = O(n)$

لأنه آخر عنصر لم يتم مقارنته

merge sort

$T(n) = 2T\left(\frac{n}{2}\right) + (n-1) \rightarrow O(n)$

نلاحظ انه تم استدعاء الدالة مرتين داخل نفسها

$\therefore T(n) = 2T\left(\frac{n}{2}\right) + O(n)$

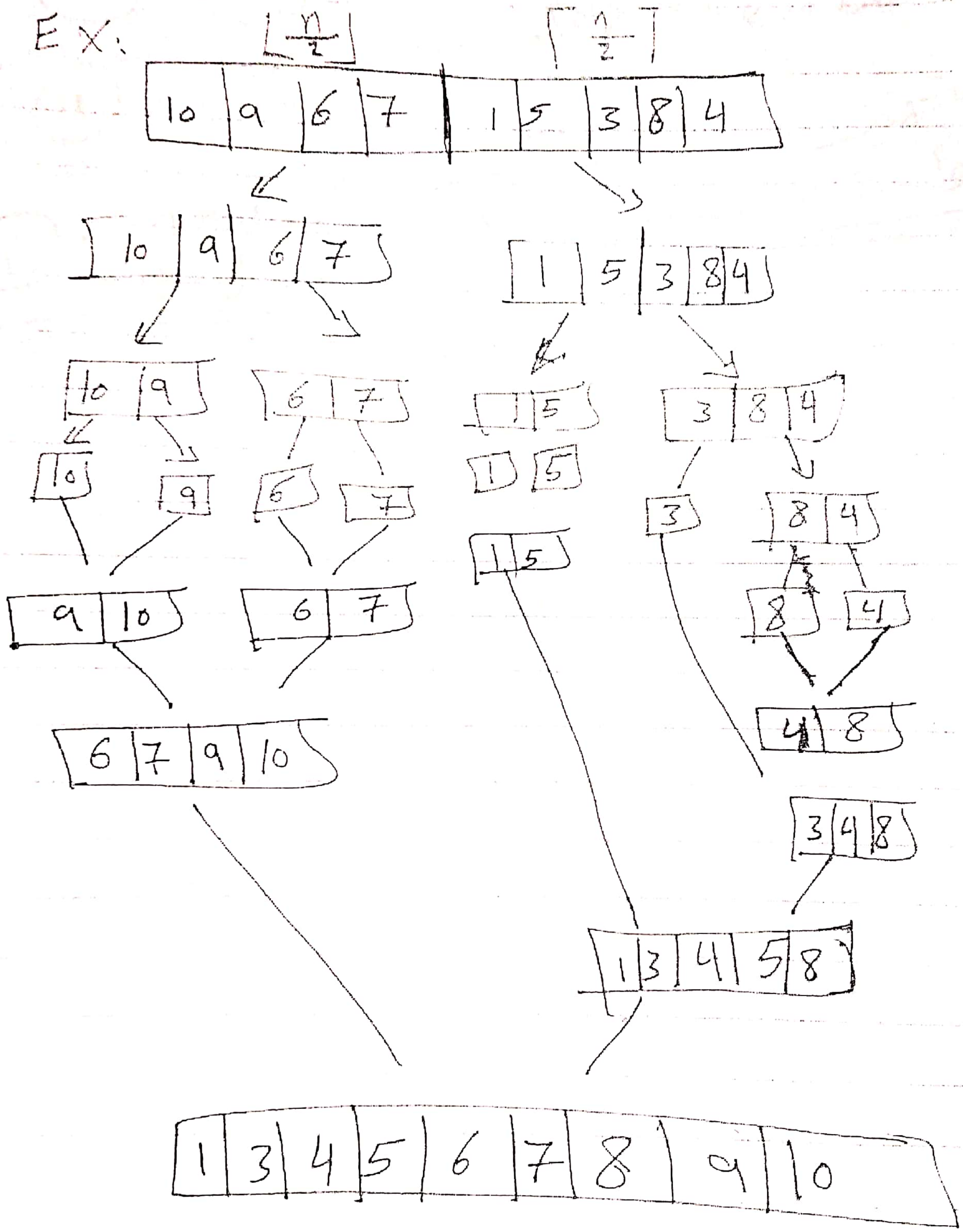
$= O(n \log n)$

نحلها باستخدام Master Theorem

مثال مهم

Cont 2:

Ex:



Example solve

$$T(n) = 2T\left(\frac{n}{3}\right) + cn^2$$

(i) using substitution method

(ii) " master Theorem

sol

الجهير ضفوا مساعدة للحل

(i) substitution method

$$T(n) = 2T\left(\frac{n}{3}\right) + cn^2$$

$$T\left(\frac{n}{3}\right) = 2T\left(\frac{n}{3^2}\right) + c\cdot\left(\frac{n}{3}\right)^2$$

$$T\left(\frac{n}{3^2}\right) = 2\cdot T\left(\frac{n}{3^3}\right) + c\cdot\left(\frac{n}{3^2}\right)^2$$

$$T\left(\frac{n}{3^3}\right) = 2T\left(\frac{n}{3^4}\right) + c\cdot\left(\frac{n}{3^3}\right)^2$$

$$T(n) = 2\boxed{T\left(\frac{n}{3}\right)} + cn^2 \longrightarrow k=1$$

$$T(n) = 2\left(2\cdot T\left(\frac{n}{3^2}\right) + c\left(\frac{n}{3}\right)^2\right) + cn^2$$

$$T(n) = 2^2\cdot\boxed{T\left(\frac{n}{3^2}\right)} + 2c\left(\frac{n}{3}\right)^2 + cn^2 \longrightarrow k=2$$

$$T(n) = 2^2\left(2\cdot T\left(\frac{n}{3^3}\right) + c\cdot\left(\frac{n}{3^2}\right)^2\right) + 2c\left(\frac{n}{3}\right)^2 + cn^2$$

$$T(n) = 2^3\cdot T\left(\frac{n}{3^3}\right) + 2^2 c\left(\frac{n}{3^2}\right)^2 + 2^1 c\left(\frac{n}{3}\right)^2 + cn^2$$

$$= 2^3\cdot T\left(\frac{n}{3^3}\right) + c\cdot n^2\left(2^2\cdot\frac{1}{(3^2)^2} + 2^1\cdot\frac{1}{(3^2)^1} + 2^0\cdot\frac{1}{(3^2)^0}\right)$$

$$= 2^3\cdot T\left(\frac{n}{3^3}\right) + c\cdot n^2\left(\left(\frac{2}{3^2}\right)^2 + \left(\frac{2}{3^2}\right)^1 + \left(\frac{2}{3^2}\right)^0\right)$$

$$= 2^3\cdot T\left(\frac{n}{3^3}\right) + cn^2\left(\left(\frac{2}{9}\right)^2 + \left(\frac{2}{9}\right)^1 + \left(\frac{2}{9}\right)^0\right)$$

$$= 2^3\cdot T\left(\frac{n}{3^3}\right) + cn^2\sum_{i=0}^{3-1}\left(\frac{2}{9}\right)^i \longrightarrow k=3$$

$$T(n) = 2^k\cdot T\left(\frac{n}{3^k}\right) + cn^2\cdot\sum_{i=0}^{k-1}\left(\frac{2}{9}\right)^i$$ (k)

Scanned by CamScanner

$\left(\frac{2}{9}\right)^i$

$$T(n) = 2^k \cdot T\left(\frac{n}{3^k}\right) + cn^2 \sum_{i=0}^{k-1} \left(\frac{2}{9}\right)^i$$

$$\leq 2^k \cdot T\left(\frac{n}{3^k}\right) + cn^2 \cdot \sum_{i=0}^{\infty} \left(\frac{2}{9}\right)^i$$

$$\leq 2^k \cdot T\left(\frac{n}{3^k}\right) + cn^2 \cdot \frac{1}{1-\frac{2}{9}} \rightarrow \frac{7}{9}$$

$$T(n) \leq 2^k \cdot T\left(\frac{n}{3^k}\right) + cn^2 \cdot \frac{9}{7}$$

$\because T(1) = c$

put $\frac{n}{3^k} = 1 \Rightarrow 3^k = n \Rightarrow k = \log_3 n$

$$\therefore T(n) \leq 2^{\log_3 n} \cdot T(1) + \frac{9c}{7} n^2$$

$$T(n) \leq n^{\log_3 2} + \frac{9c}{7} n^2$$

$$\therefore \boxed{T(n) \text{ is } O(n^2)}$$

Recurrence Relation: تمارين

solve $T(n) = 2T(\frac{n}{2}) + 3n^2$

using substitution sol

خطوات التجهيز

$T(n) = 2T(\frac{n}{2}) + 3n^2$

$\boxed{T(\frac{n}{2}) = 2T(\frac{n}{2^2}) + 3 \cdot (\frac{n}{2})^2}$

$\boxed{T(\frac{n}{2^2}) = 2T(\frac{n}{2^3}) + 3 \cdot (\frac{n}{2^2})^2}$

خطوات الحل الرئيسية

$T(n) = 2T(\frac{n}{2}) + 3n^2 \longrightarrow (1)$

$T(n) = 2\left(2 \cdot T(\frac{n}{2^2}) + 3 \cdot (\frac{n}{2})^2\right) + 3n^2$

$= 2^2 \cdot T(\frac{n}{2^2}) + 3 \cdot 2 \cdot (\frac{n}{2})^2 + 3 \cdot n^2$

$= 2^2 \cdot T(\frac{n}{2^2}) + 3 \cdot 2 \cdot (\frac{1}{2})^2 \cdot n^2 + 3 \cdot n^2$

$T(n) = 2^2 \cdot T(\frac{n}{2^2}) + 3 \cdot n^2 \cdot (\frac{1}{2})^1 + 3n^2 \cdot (\frac{1}{2})^0 \longrightarrow (2)$

$T(n) = 2^2\left(2 \cdot T(\frac{n}{2^3}) + 3 \cdot (\frac{n}{2^2})^2\right) + 3n^2 \cdot (\frac{1}{2})^1 + 3n^2 (\frac{1}{2})^0$

$= 2^3 \cdot T(\frac{n}{2^3}) + 3 \cdot n^2 \cdot 2^2 \cdot (\frac{1}{2^2})^2 + 3n^2 (\frac{1}{2})^1 + 3n^2 \cdot (\frac{1}{2})^0$

$T(n) = 2^3 \cdot T(\frac{n}{2^3}) + 3n^2 \cdot (\frac{1}{2})^2 + 3n^2 \cdot (\frac{1}{2})^1 + 3n^2 \cdot (\frac{1}{2})^0 \longrightarrow (3)$

$$T(n) = 2^k \cdot T\left(\frac{n}{2^k}\right) + 3n^2 \cdot \left(\frac{1}{2}\right)^{k-1} + \cdots + 3n^2 \cdot \left(\frac{1}{2}\right)^0$$

$$T(n) = 2^k \cdot T\left(\frac{n}{2^k}\right) + 3n^2 \cdot \sum_{i=0}^{k-1} \left(\frac{1}{2}\right)^i$$

$$= 2^k T\left(\frac{n}{2^k}\right) + 3n^2 \cdot \frac{1-\left(\frac{1}{2}\right)^k}{1-\frac{1}{2}}$$

$$\frac{5-1}{6-2} = \frac{1-5}{2-6}$$

$$\leq 2$$

التوضيح أسفل

$$T(n) \leq 2^k T\left(\frac{n}{2^k}\right) + 3n^2 \cdot 2$$

$$T(n) \leq 2^k T\left(\frac{n}{2^k}\right) + 6n^2$$

put $\left(\frac{n}{2^k}\right) = 1 \Rightarrow 2^k = n \Rightarrow k = \log_2 n$

$$\therefore T(n) \leq n\,T(1) + 6n^2$$

$$\leq c \cdot n + \underbrace{6n^2}_{\text{overhead}} = O(n^2)$$

قاعدة هامة

$$\sum_{i=0}^{k} \left(\frac{1}{2}\right)^i \leq \sum_{i=0}^{\infty} \left(\frac{1}{2}\right)^i = \frac{1}{1-\frac{1}{2}}$$

$$= \boxed{2}$$

توضيح للسبب

$$\left(\frac{1}{2}\right)^0 + \left(\frac{1}{2}\right)^1 + \left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^3 + \left(\frac{1}{2}\right)^4 + \left(\frac{1}{2}\right)^5$$

Ex(1) recursive Tree

$T(n) = 2T(\frac{n}{2}) + 4n \quad , T(1) = 4$

	#nodes		Row sum
$T(\frac{n}{2^0})$	1	$4n$	$4n$
$T(\frac{n}{2^1})$	2	$4\frac{n}{2} \quad 4\frac{n}{2}$	$4n$
$T(\frac{n}{4}) = T(\frac{n}{2^2})$	4	$4\frac{n}{2^2} \quad 4\frac{n}{2^2} \quad 4\frac{n}{2^2} \quad 4\frac{n}{2^2}$	$4n$
$T(\frac{n}{2^3})$	8	$4\frac{n}{2^3} \quad 4\frac{n}{2^3} \quad 4\frac{n}{2^3} \quad 4\frac{n}{2^3} \quad \dots \quad \frac{4n}{2^3}$	$4n$
$T(\frac{n}{2^k})$	2^k	$\frac{4n}{2k} \quad \dots \quad \frac{4n}{2k}$	$\rightarrow 4n$

$\because T(1) = 4 \quad \therefore \frac{n}{2k} = 1 \Rightarrow k = \log n$

$\therefore \text{Total} = \sum_{i=0}^{\log_2 n} 4n = \underbrace{4n + 4n + \dots + 4n}_{\log_2 n + 1 \text{ c/s}}$

$= 4n(\log n + 1) = 4n\log n + 4n$

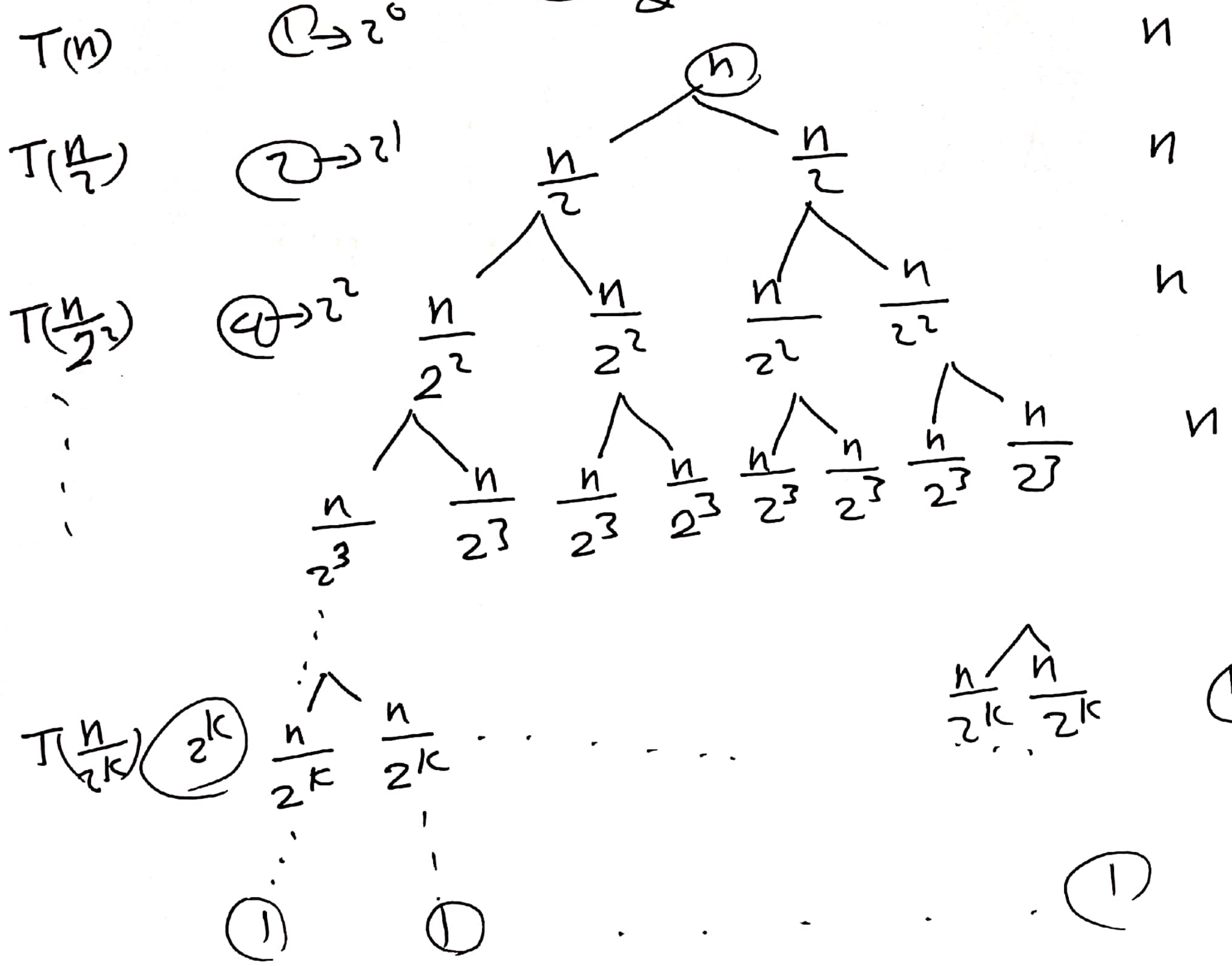
$\therefore \Theta(n \log n)$

05087973247 idealele

Ex

$$T(n) = 2T(n/2) + n$$

حل



$$\frac{n}{2^k} = 1 \Rightarrow 2^k = n \Rightarrow k = \log n$$

$$\therefore T(n) = \underbrace{n + n + \cdots + n}_{\log n + 1} = n(\log n + 1)$$

عددهم

$$\simeq n\log n + n$$

$$O(n \log n)$$

0508793247

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\frac{n}{b}\right) + cn & n > 1 \end{cases}$$

- So…

$$T(n) = \begin{cases} \Theta(n) & a < b \\ \Theta(n \log_b n) & a = b \\ \Theta\left(n^{\log_b a}\right) & a > b \end{cases}$$

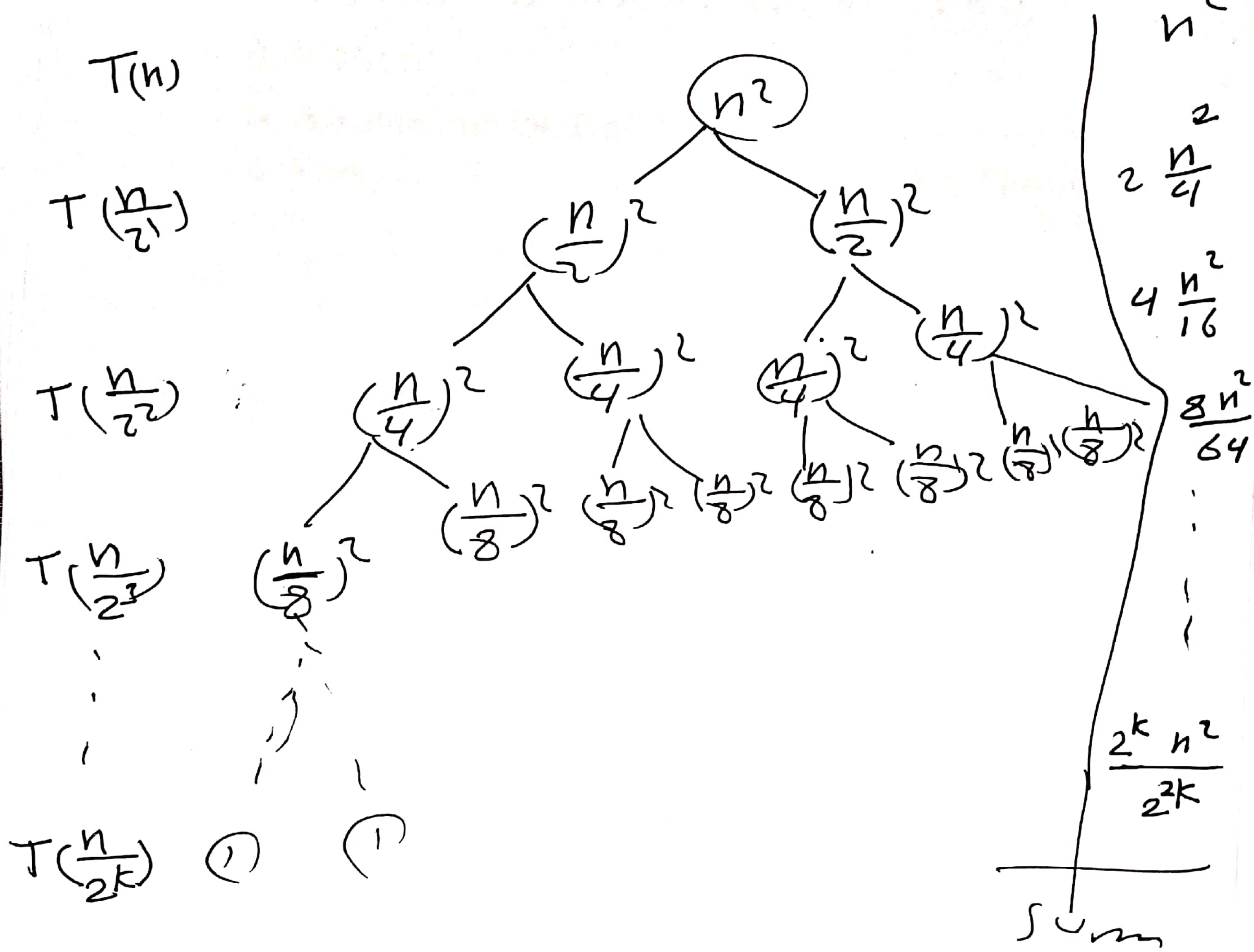
Substitution Method

- Solution for $T(n) = 2T(n/2) + n$

Guess a solution,
Verify its correctness

Recursive Tree Ex 2

$$T(n) = 2T\left(\frac{n}{2}\right) + n^2$$



$$\text{Sum} = n^2 + \frac{n^2}{2} + \frac{n^2}{4} + \frac{n^2}{8} + \cdots + \frac{n^2}{2^k}$$

$$= n^2\left(1 + \frac{1}{2} + \frac{1}{2^2} + \frac{1}{2^3} + \cdots + \frac{1}{2^k}\right)$$

$$= n^2 \cdot \frac{\left(\frac{1}{2}\right)^{k+1} - 1}{\left(\frac{1}{2}\right) - 1} \leq n^2 \cdot \frac{1}{1 - \frac{1}{2}} = 2n^2$$

$$\boxed{O(n^2)}$$

Substitution Method

- Guess $T(n) = O(n \lg n)$ i.e, $T(n) \leq c\, n \lg n$
- Induction:
  - Assume true for T(n/2)
  - $T(n) = 2\,T(n/2) + n$

$$\leq 2\,c\,n/2 \lg (n/2) + n$$
$$= c\,n \lg n - (c-1)\,n$$
$$\leq c\,n \lg n \qquad \text{if } c > 1$$

Verification

- Base case:
  - $T(1)$ .... no
  - But true for $T(2)$ for sufficiently large c

Solve c

The Master Theorem

- Given: a *divide and conquer* algorithm
  - An algorithm that divides the problem of size n into a subproblems, each of size n/b
  - Let the cost of each stage (i.e., the work to divide the problem + combine solved subproblems) be described by the function $f(n)$
- Then, the Master Theorem gives us a cookbook for the algorithm's running time:

The Master Theorem

- if T(n) = aT(n/b) + f(n) then

$$T(n) = \begin{cases} \Theta\left(n^{\log_b a}\right) & f(n) = O\left(n^{\log_b a-\varepsilon}\right) \\ \Theta\left(n^{\log_b a} \log n\right) & f(n) = \Theta\left(n^{\log_b a}\right) \\ \Theta\left(f(n)\right) & f(n) = \Omega\left(n^{\log_b a+\varepsilon}\right) \text{AND} \\ & af(n/b) < cf(n) \text{ for large } n \end{cases} \left.\vphantom{\begin{matrix} a\\a\\a\\a\\a\\a \end{matrix}}\right\} \begin{matrix} \varepsilon > 0 \\ c < 1 \end{matrix}$$

The Master Method

- The Master method applies to recurrences of the form

  $$T(n) = a\,T(n/b) + f(n)$$

  where $a \geq 1$ and $b > 1$, and $f(n)$ is an asymptotically positive function.

Case One

- $f(n) = O(n^{\log_b a - \varepsilon})$ for some const. $\varepsilon > 0$
  then , $T(n) = \Theta(n^{\log_b a})$
  - $f(n)$ grows polynomially slower than $n^{\log_b a}$ (by n^{ε})
  - the summation of $f(n)$ from each levels in recursion tree is consumed by n^{ε}

Case Two

- $f(n) = \Theta(n^{\log_b a})$ for some const. $\varepsilon > 0$
  then , $T(n) = \Theta(n^{\log_b a} \lg n)$

Case Three

- $f(n) = \Omega(n^{\log_b a + \varepsilon})$ for some const. $\varepsilon > 0$
  and if $af(n/b) \leq cf(n)$ for all sufficiently large n,
  then, $T(n) = \Theta(f(n))$.

Note that the three cases are not complete.
There are gaps among them.

Other Examples

MergeSort ($A, 1, n$)

$m = n / 3;$

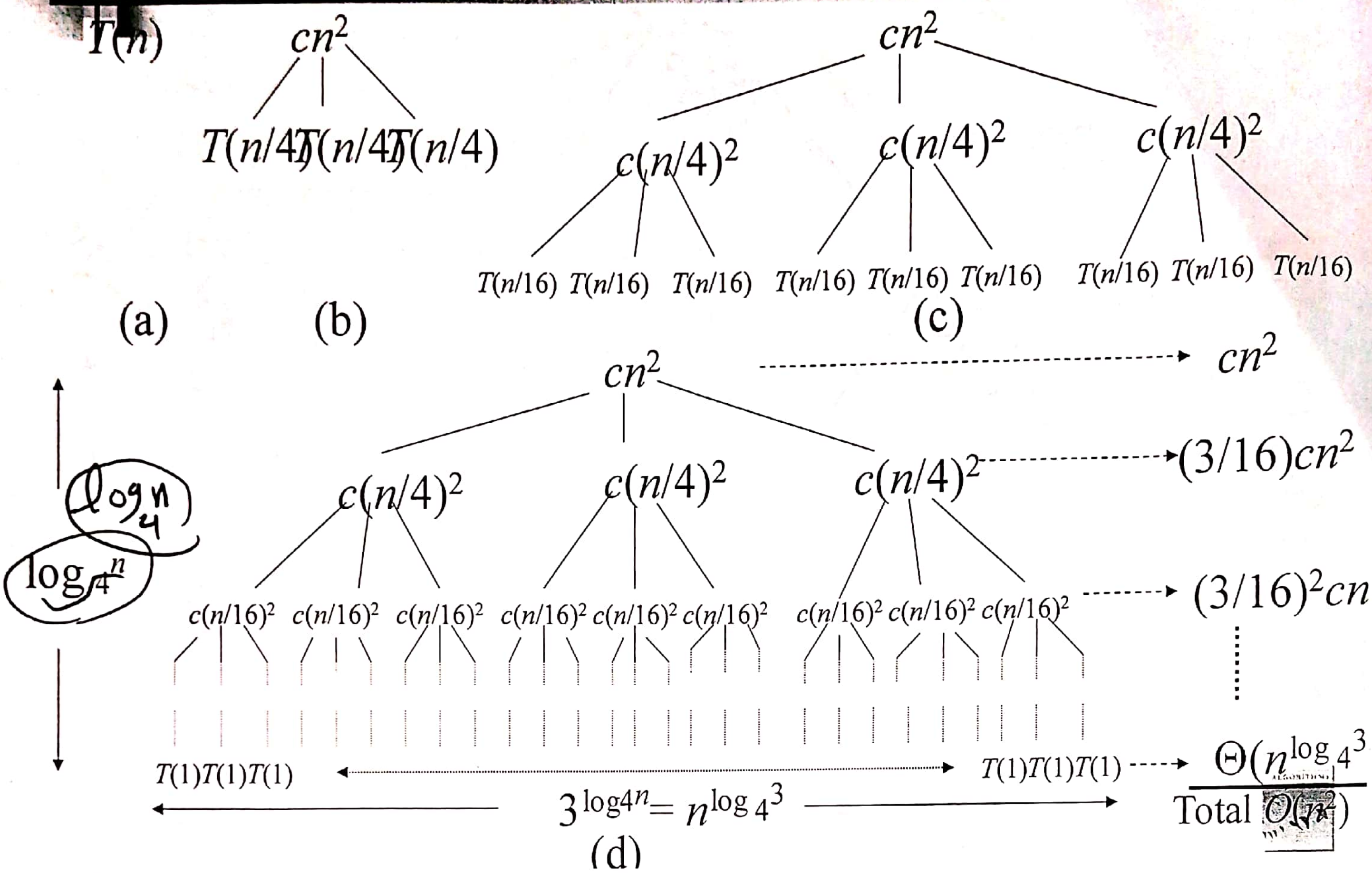
$A1 =$ MergeSort ($A, 1, m$);

3-way merge sort: $T(n) = 3T(n/3) + n$

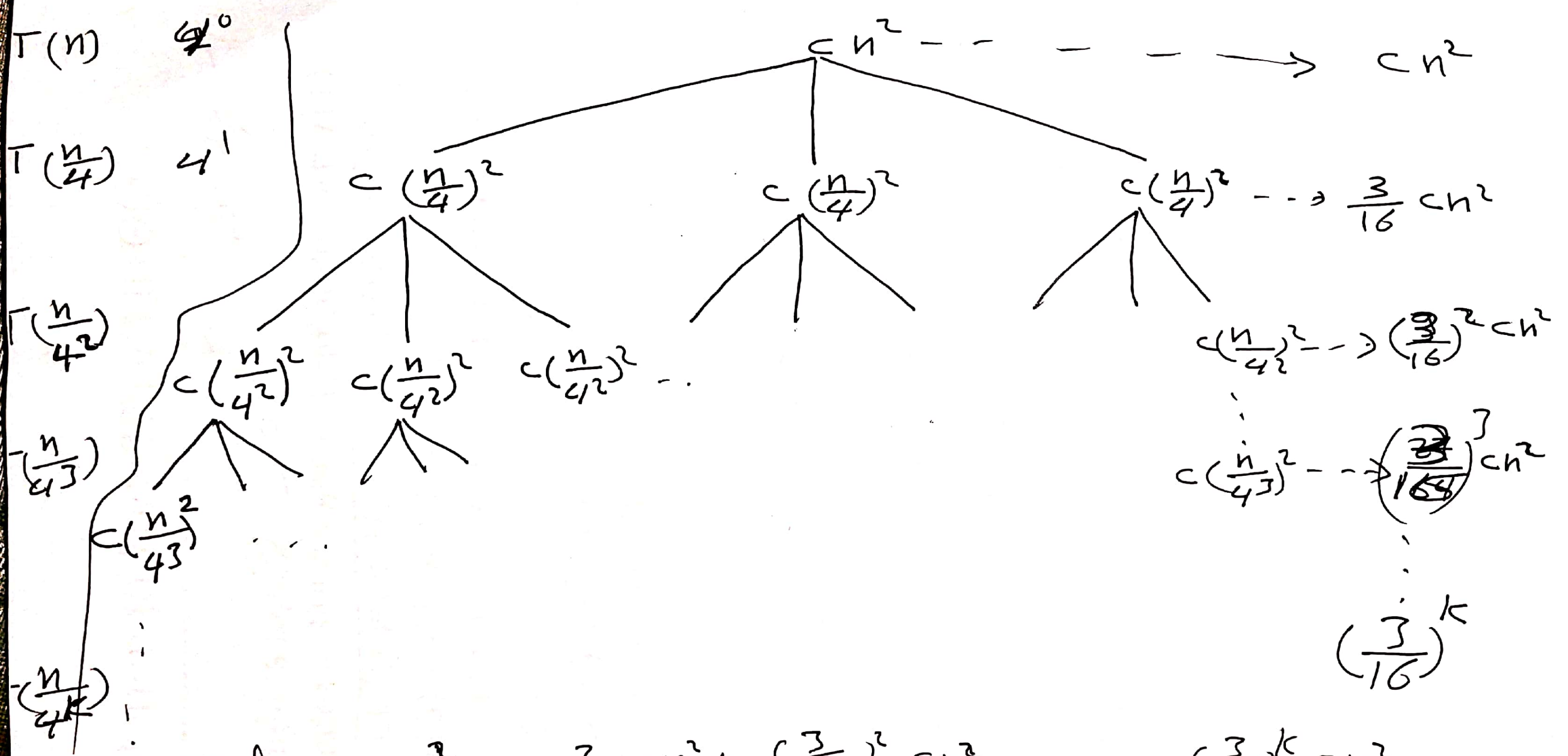
Merge ($A1, A2$);

$$T(n) = T(n/3) + T(2n/3) + n$$

Recursion Tree for $T(n)=3T(\lfloor n/4 \rfloor)+\Theta(n^2)$



EX: $T(n) = 3T\left(\frac{n}{4}\right) + \Theta(n^2) = 3T\left(\frac{n}{4}\right) + cn^2$



$$\text{Total} = cn^2 + \frac{3}{16}cn^2 + \left(\frac{3}{16}\right)^2 cn^2 + \cdots + \left(\frac{3}{16}\right)^k cn^2$$

$$= cn^2\left(1 + \frac{3}{16} + \cdots + \left(\frac{3}{16}\right)^k\right) \leq cn^2\left(1 + \frac{3}{16} + \cdots \infty\right)$$

$$= cn^2 \frac{1}{1 - \frac{3}{16}} = \frac{16}{13}cn^2 = O(n^2)$$

Scanned by CamScanner

Solution to $T(n)=3T(\lfloor n/4 \rfloor)+\Theta(n^2)$

3^k هدا الشجرة السابقة

- The height is $\log_4 n$
- #leaf nodes $= 3^{\log_4 n} = n^{\log_4 3}$. Leaf node cost: $T(1)$.
- Total cost $T(n)=cn^2+(3/16)\,cn^2+(3/16)^2\,cn^2+$
  $\cdots +(3/16)^{\log_4 n-1}\,cn^2+\Theta(n^{\log_4 3})$
  $=(1+3/16+(3/16)^2+\cdots+(3/16)^{\log_4 n-1})\,cn^2+\Theta(n^{\log_4 3})$
  $<(1+3/16+(3/16)^2+\cdots+(3/16)^m+\cdots)\,cn^2+\Theta(n^{\log_4 3})$
  $=(1/(1-3/16))\,cn^2+\Theta(n^{\log_4 3})$
  $=16/13cn^2+\Theta(n^{\log_4 3})$
  $=O(n^2)$.

هدا الشجرة في الصفحة السابقة

$$\frac{n}{4^k}=1 \Rightarrow 4^k=n \Rightarrow k=\log_4 n$$

وهدا هو ارتفاع الشجرة

$$\#\text{leaf nodes} = 3^k = 3^{\log_4 n} = n^{\log_4 3}$$

Recursion Tree of
$T(n)=T(n/3)+ T(2n/3)+O(n)$

$T(n)$

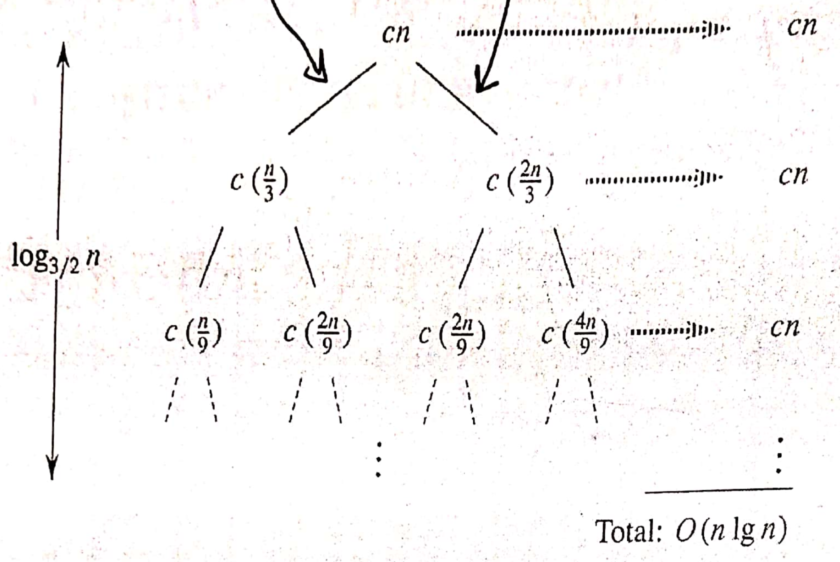


Figure 4.2 A recursion tree for the recurrence $T(n) = T(n/3) + T(2n/3) + cn$.

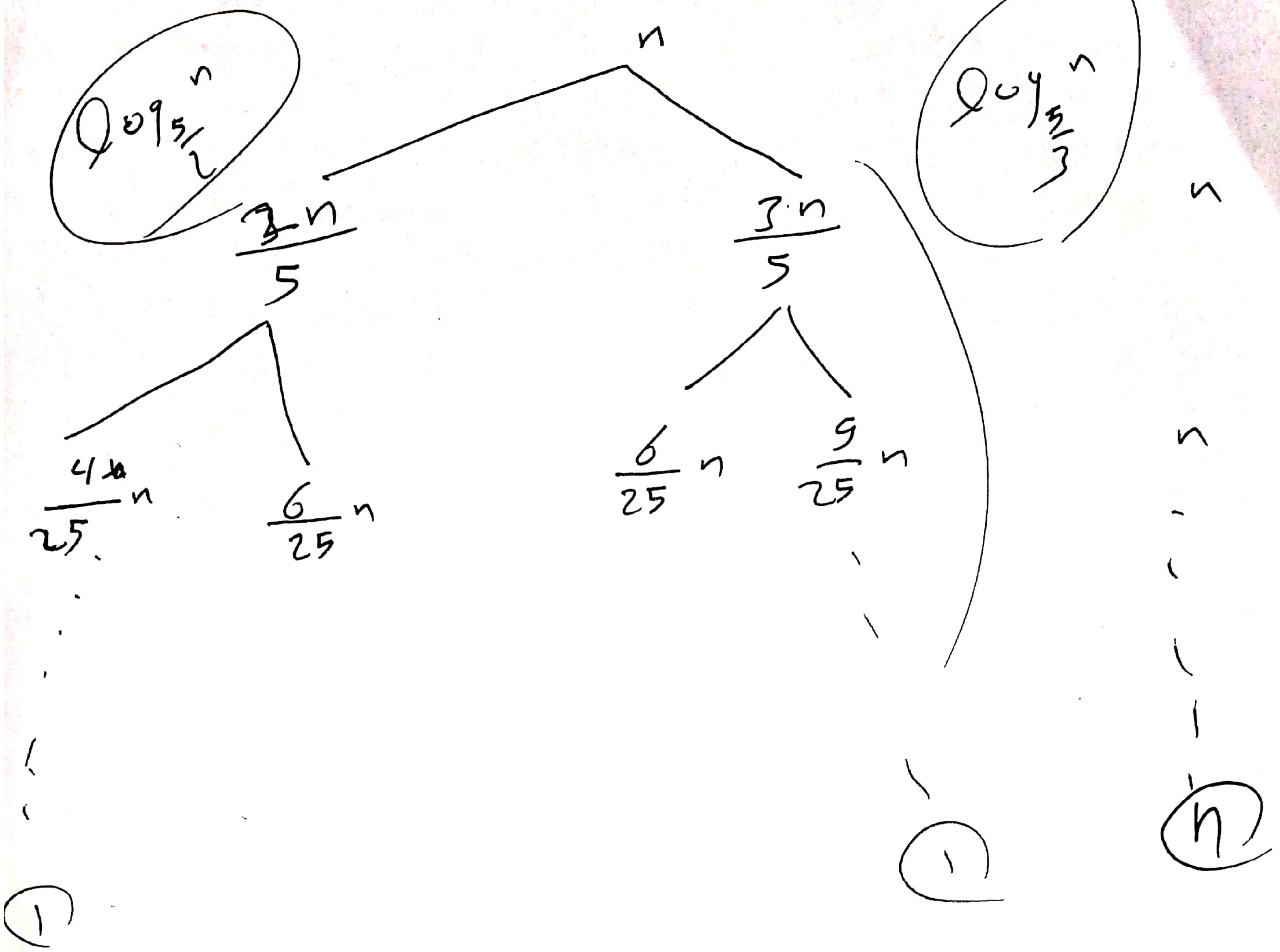
$\frac{n}{3}$, $\frac{2n}{3}$

One more example

- $T(n)=T(n/3)+T(2n/3)+O(n)$.
- Construct its recursive tree
- $T(n)=O(cn\lg_{3/2}{}^{n}) = O(n\lg n)$.
- Prove $T(n) \leq dn\lg n$.

4

$$T(n) = T\left(\frac{2n}{5}\right) + T\left(\frac{3n}{5}\right) + n$$



$$\boxed{T(n) = cn \log_{5/3} n}$$

$32^{1/n}$

$32^{n/n}$

$\frac{1}{4}$ طويل

$\frac{1}{2}$ قصير